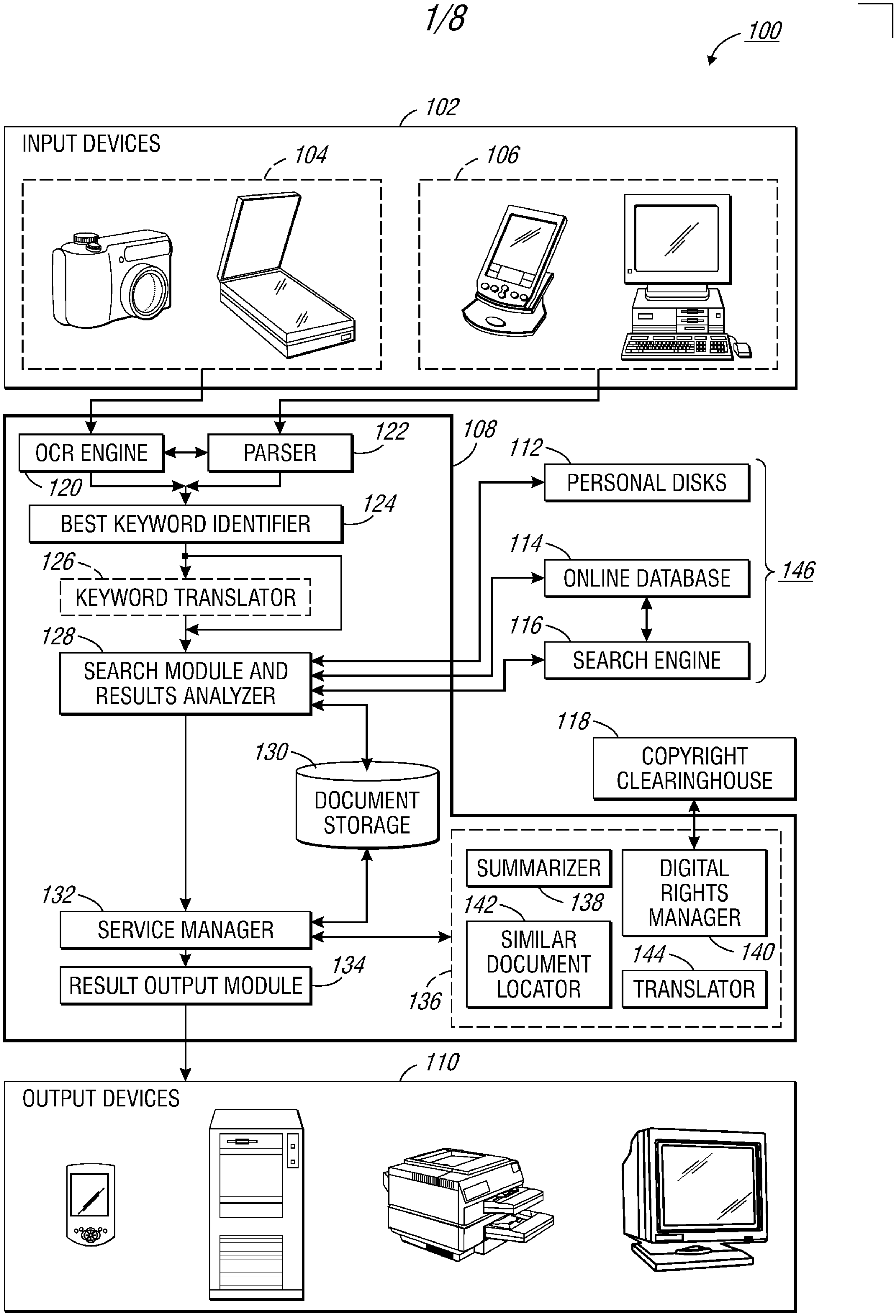

100
102
INPUT DEVICES
104
106
OCR ENGINE
PARSER
122
108
120
112
PERSONAL DISKS
BEST KEYWORD IDENTIFIER
124
126
KEYWORD TRANSLATOR
114
ONLINE DATABASE
146
128
116
SEARCH ENGINE
SEARCH MODULE AND RESULTS ANALYZER
130
118
COPYRIGHT CLEARINGHOUSE
DOCUMENT STORAGE
SUMMARIZER
DIGITAL RIGHTS MANAGER
138
142
132
SERVICE MANAGER
SIMILAR DOCUMENT LOCATOR
140
144
TRANSLATOR
134
RESULT OUTPUT MODULE
136
110
OUTPUT DEVICES

FIG. 1

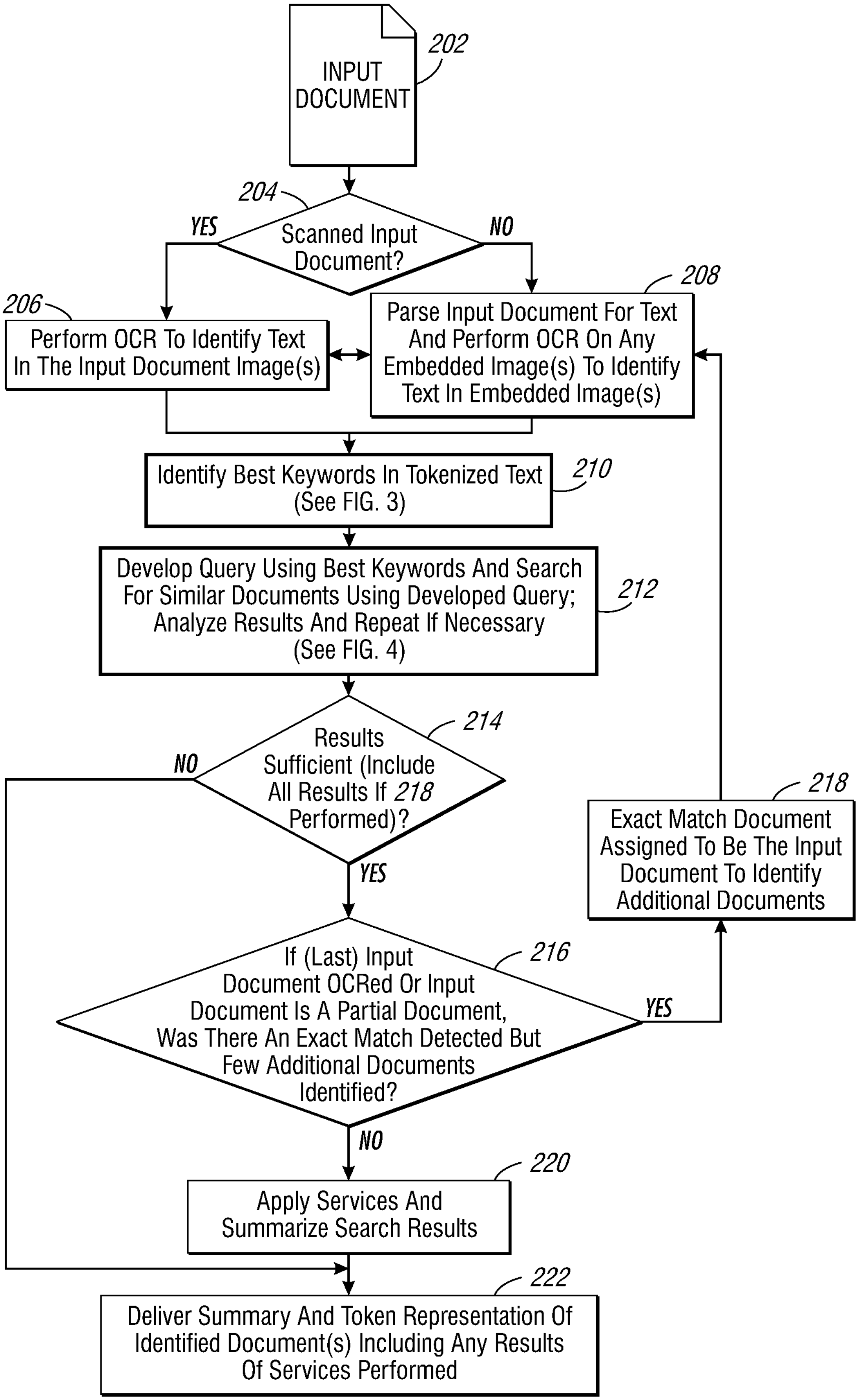

INPUT DOCUMENT
202
204
Scanned Input Document?
YES
NO
206
Perform OCR To Identify Text In The Input Document Image(s)
208
Parse Input Document For Text And Perform OCR On Any Embedded Image(s) To Identify Text In Embedded Image(s)
Identify Best Keywords In Tokenized Text (See FIG. 3)
210
Develop Query Using Best Keywords And Search For Similar Documents Using Developed Query; Analyze Results And Repeat If Necessary (See FIG. 4)
212
214
Results Sufficient (Include All Results If 218 Performed)?
NO
YES
218
Exact Match Document Assigned To Be The Input Document To Identify Additional Documents
216
If (Last) Input Document OCRed Or Input Document Is A Partial Document, Was There An Exact Match Detected But Few Additional Documents Identified?
YES
NO
220
Apply Services And Summarize Search Results
222
Deliver Summary And Token Representation Of Identified Document(s) Including Any Results Of Services Performed

FIG. 2

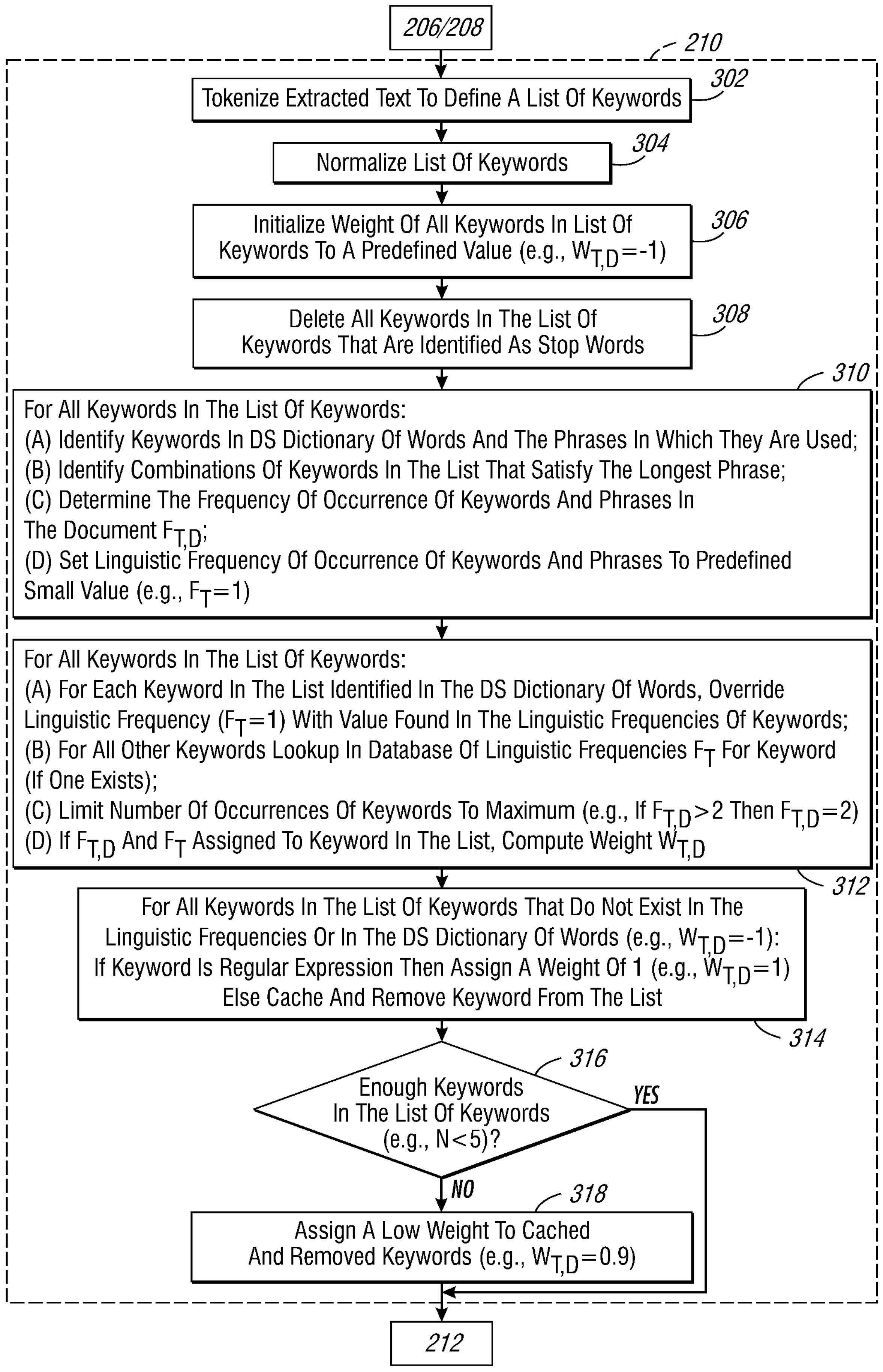

206/208
210
Tokenize Extracted Text To Define A List Of Keywords
302
Normalize List Of Keywords
304
Initialize Weight Of All Keywords In List Of Keywords To A Predefined Value (e.g., $W_{T,D}$=-1)
306
Delete All Keywords In The List Of Keywords That Are Identified As Stop Words
308
310
For All Keywords In The List Of Keywords:
(A) Identify Keywords In DS Dictionary Of Words And The Phrases In Which They Are Used;
(B) Identify Combinations Of Keywords In The List That Satisfy The Longest Phrase;
(C) Determine The Frequency Of Occurrence Of Keywords And Phrases In The Document $F_{T,D}$;
(D) Set Linguistic Frequency Of Occurrence Of Keywords And Phrases To Predefined Small Value (e.g., F_T=1)
For All Keywords In The List Of Keywords:
(A) For Each Keyword In The List Identified In The DS Dictionary Of Words, Override Linguistic Frequency (F_T=1) With Value Found In The Linguistic Frequencies Of Keywords;
(B) For All Other Keywords Lookup In Database Of Linguistic Frequencies F_T For Keyword (If One Exists);
(C) Limit Number Of Occurrences Of Keywords To Maximum (e.g., If $F_{T,D}$>2 Then $F_{T,D}$=2)
(D) If $F_{T,D}$ And F_T Assigned To Keyword In The List, Compute Weight $W_{T,D}$
312
For All Keywords In The List Of Keywords That Do Not Exist In The Linguistic Frequencies Or In The DS Dictionary Of Words (e.g., $W_{T,D}$=-1):
If Keyword Is Regular Expression Then Assign A Weight Of 1 (e.g., $W_{T,D}$=1)
Else Cache And Remove Keyword From The List
314
316
Enough Keywords In The List Of Keywords (e.g., N<5)?
YES
NO
318
Assign A Low Weight To Cached And Removed Keywords (e.g., $W_{T,D}$=0.9)
212

FIG. 3

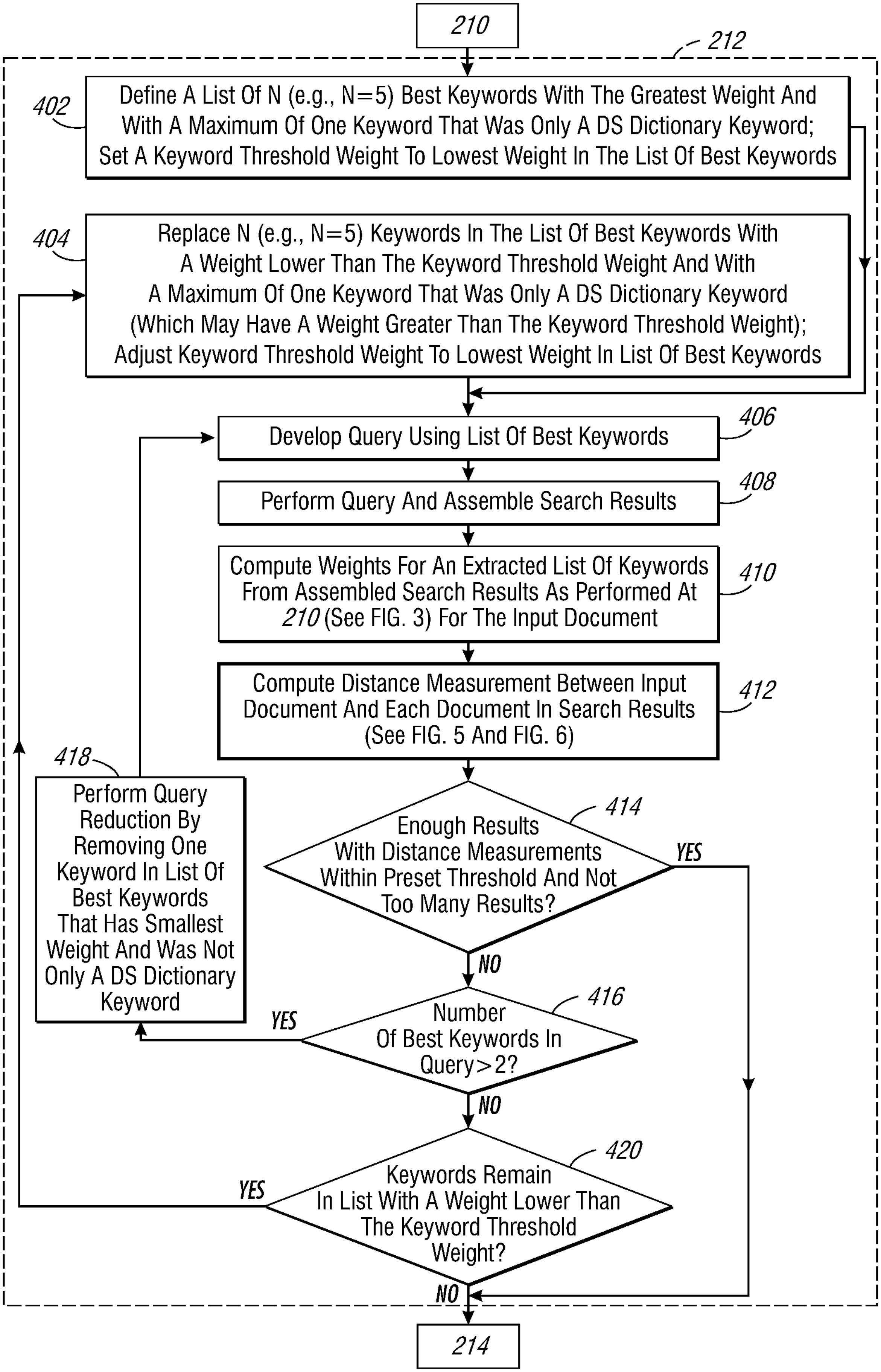

210
212
402
Define A List Of N (e.g., N=5) Best Keywords With The Greatest Weight And With A Maximum Of One Keyword That Was Only A DS Dictionary Keyword; Set A Keyword Threshold Weight To Lowest Weight In The List Of Best Keywords
404
Replace N (e.g., N=5) Keywords In The List Of Best Keywords With A Weight Lower Than The Keyword Threshold Weight And With A Maximum Of One Keyword That Was Only A DS Dictionary Keyword (Which May Have A Weight Greater Than The Keyword Threshold Weight); Adjust Keyword Threshold Weight To Lowest Weight In List Of Best Keywords
Develop Query Using List Of Best Keywords
406
Perform Query And Assemble Search Results
408
Compute Weights For An Extracted List Of Keywords From Assembled Search Results As Performed At 210 (See FIG. 3) For The Input Document
410
Compute Distance Measurement Between Input Document And Each Document In Search Results (See FIG. 5 And FIG. 6)
412
418
Perform Query Reduction By Removing One Keyword In List Of Best Keywords That Has Smallest Weight And Was Not Only A DS Dictionary Keyword
414
Enough Results With Distance Measurements Within Preset Threshold And Not Too Many Results?
YES
NO
416
Number Of Best Keywords In Query>2?
YES
NO
420
Keywords Remain In List With A Weight Lower Than The Keyword Threshold Weight?
YES
NO
214

FIG. 4

```
CALCULATE_SIMILARITY [D1,D2] -calculates the similarity between documents D1 and D2
•Input:
    • D1: List of keywords of the input document
    • D2: List of keywords of document from search results
•Output:
    • Similarity "S": The computed similarity between D1 and D2
{
502 For the document D1, calculate with the keyword weights of D1:
        • unique attributes sum: Sum1 = the sum of the weights of keywords in D1 that do not appear in D2
        • total sum: Sum2 = the sum of the weights of keywords in D1
        • shared sum: Sum3 = the sum of the weights of keywords in D1 that also appear in D2
        • ratio: R = (the number of keywords in D1 not in D2)/(the number of keywords in D1)

504 For the document D2, calculate with the keyword weights of D2:
        • unique attributes sum: Sum4 = the sum of the weights of  keywords in D2 that do not appear in D1
        • shared sum: Sum5 = the sum of weights of keywords in D2 that also appear in D1

506 If D1 originates from a hardcopy document, calculate the tolerance ratio "T":
        K = a constant defined by OCR error rate at the keyword level, if no OCR error is detected, K is set to 0
        T = K* (Sum2 – Sum3)/(Sum2)

508 Calculate the inclusion ratio "I" (i.e., percentage of keywords from D1 that are in D2):
        I = (Sum3)/(Sum2) + T
```

TO FIG. 6

FIG. 5

FROM FIG. 5

```
     {If I>90% (i.e., if an inclusion is detected, e.g., 90 % of the keywords from D1 are in D2):
           Sum6 = Ordered_Sum [D1,D2] - sum of the weights of keywords in D1
                                        with same neighbors in D2 (SEE FIG. 7)
         • Calculate ordered inclusion ratio "I2":
                I2 = (Sum6)/(Sum2)
510             if (I2>I) then S = I
                else
                     if (D1 originates from a hardcopy document and I2>50%)
                        if (R<20 %) then S = I else S = I2
                     else S = I2
     {else (i.e., if no inclusion is detected):
         • Calculate the Jaccard similarity distance measure:
                Sum7 = Sum1 + Sum4  + Sum5
                S = (Sum5) / (Sum7)
         • If S>90% : (a revision is detected,  i.e., Jaccard similarity S>90 %;
                     otherwise a related document may be detected)
                • Sum8 = Ordered_Sum [D2,D1] - sum of the weights of keywords from D2
                                                with same neighbors in D1
                • Calculate ordered similarity:
512                       S2 = (Sum8) / (Sum1 + Sum4 + Sum8)
                if (S2 > S)
                          if (D1 originates from a hardcopy document)
                                 if (S2>50%)
                                        if (R>20%) then  S = S1
                                 else S = S1
                          else S = S1
                else  S = S1
     }
```

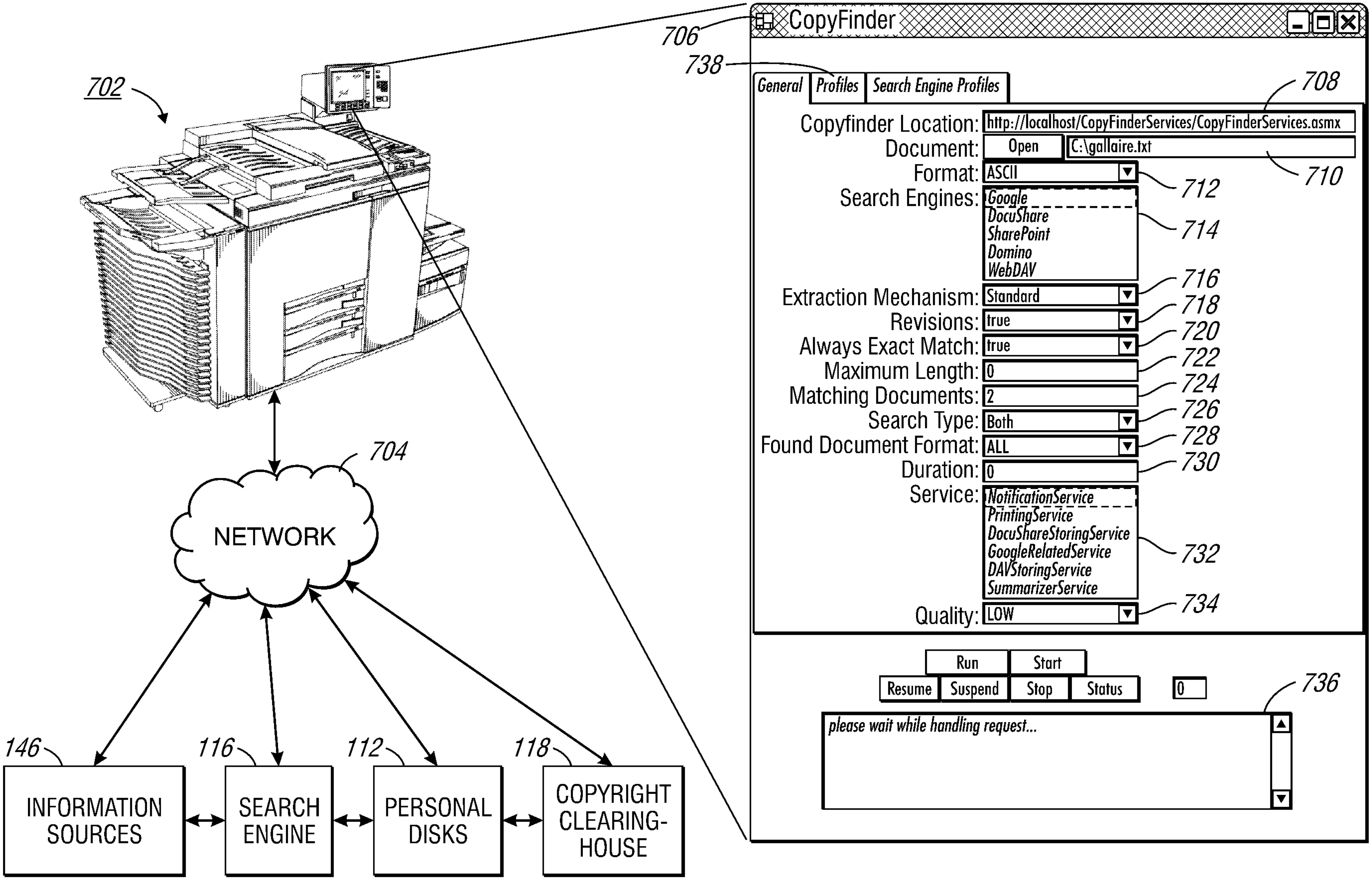
FIG. 6

```
     Ordered_Sum [L1, L2] - calculates sum of the weights of keywords from L1 with same neighbors in L2
     • Input:
          • L1: List 1
          • L2: List 2
     • Output
          • Sum: the ordered sum
     {
600  Define the tolerance "T" minimal percentage for neighbors :
     T = K*50%, where K depends on the OCR error at the keyword level

     for each term t (i.e., keyword) of L1:
        identify all possible positions Pi of term t in L2
        if (t exists in L2)
             identify N neighbors on both sides of term t in L1
             (by default N=5 and depends on the position of the term t in the L1)

             for each position Pi of term t found in L2:
602                 if the position Pi of the term t is at a limit of L2: (Pi<N or Pi>(L2 size-N))
                         increase the ordered sum with the weight of term t in L1: Sum += Wt
                    else
                         identify N neighbors on both sides of term t in L2
606                      Calculate the percentage of common neighbors "C" of term t between L1 and L2
                         if ( C > 80% - T)    increase the ordered sum with the weight of term t in L1: Sum += Wt
        else
604          increase the ordered sum with the weight of term t in L1: Sum += Wt
     }
```

FIG. 7



CopyFinder

General | Profiles | Search Engine Profiles

Copyfinder Location: http://localhost/CopyFinderServices/CopyFinderServices.asmx

Document: Open C:\gallaire.txt

Format: ASCII

Search Engines: Google, DocuShare, SharePoint, Domino, WebDAV

Extraction Mechanism: Standard

Revisions: true

Always Exact Match: true

Maximum Length: 0

Matching Documents: 2

Search Type: Both

Found Document Format: ALL

Duration: 0

Service: NotificationService, PrintingService, DocuShareStoringService, GoogleRelatedService, DAVStoringService, SummarizerService

Quality: LOW

0

please wait while handling request...

FIG. 8